Stroke Prediction

Team members– Shrutayu Aggarwal (B20CS096), Priyanshu Jain (B20CS044), Pratul Singh (B20CS095)

➢ Problem Statement

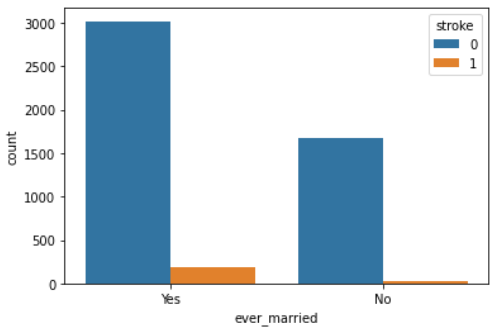
Stroke is the 2nd leading cause of death globally, responsible for approximately 11% of total deaths. The given Dataset is used to predict whether a patient is likely to get a stroke based on the input parameters like gender, age, various diseases, and smoking status.

➢ Dataset Description

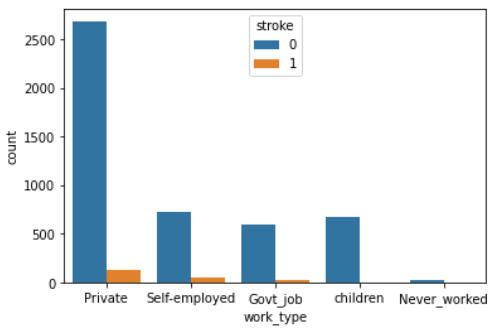
The given dataset has 12 columns which includes id, gender, age, hypertension (1 if person has hypertension otherwise 0), heart disease (1 if person has otherwise 0), whether person is married or not, work type of person, residence type of person, person's average glucose level, person's BMI, smoking status of person and the label column.

➢ Pre- processing and Exploratory Data Analysis of dataset

→ Rows with null values were dropped.
→ id column is dropped since it has no significance related with label.
→ Dataset is imbalanced as the labels for the true class are only about 4.3%.

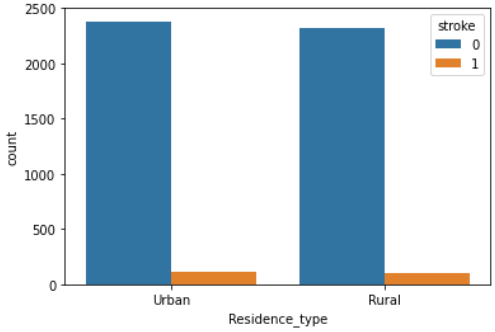


→ chance of getting stroke if married is larger than not married.

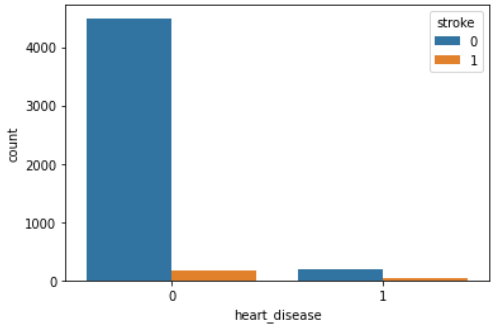


→ The people who work are likely to get stroke

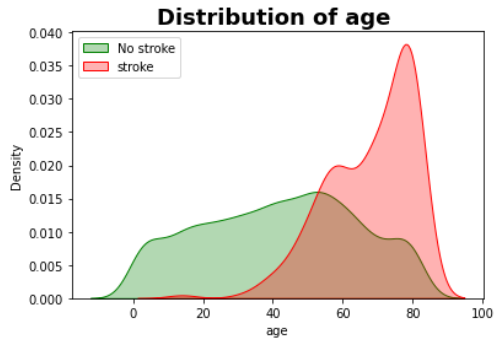
Gender having value as 'other' is dropped since there is only a single row for this.



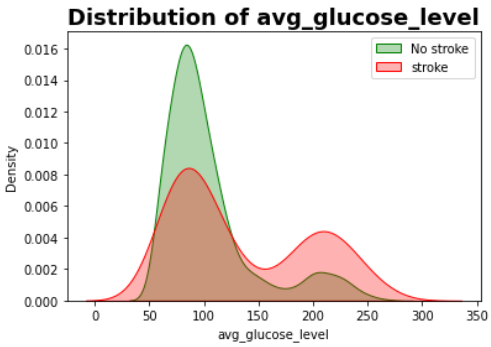
→ Residence_type classes seem to be approximately equal



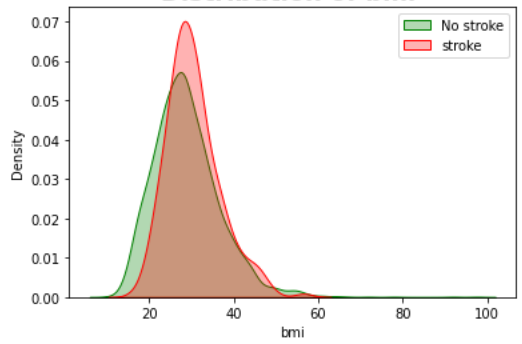
→ It seems like people not having heart disease are more likely to have stroke.



→ people with more age are likely to have stroke than the people with less age.



→ There is very little relation between avg_glucose_level and stroke. Similar glucose level has stroke as well as not stroke. If a person has avg_glucose_level > 150, he/she has very likely to have stroke.



→ There is no relation between BMI and stroke. Similar BMI has stroke as well as not stroke. Hence BMI column is dropped.

→ All the categorical columns are LabelEncoded ().

→ As we can see that continuous columns are measured in different scales so we need to standardize them by doing standardization. So, all the continuous columns are normalized using StandardScaler ().

→ Since the dataset is imbalanced so, stratified spilt is used to split the dataset in train, validation and test set in the ratio 70:10:20, to get the equal proportion of both the classes in each set.

➢ Metric Selection

Now, since the dataset is imbalanced and it is the binary classification problem, the best metric for evaluation would be Area under the ROC curve (AUC). We can also use precision and recall, but AUC combines both of these metrices.

➢ Modelling the formed dataset without taking care of Imbalance

Various classification models are fitted with the above dataset. These models include Logistic Regression, Random Forest Classifier, XGB Classifier, LGBM Classifier, Bernoulli NB, Gradient Boosting Classifier, Decision Tree, KNeighbours Classifier and the above metric is used to check which model works how. The result obtained is

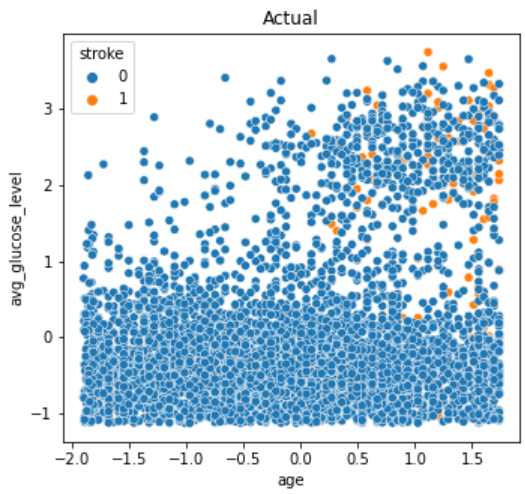
Model	Precision	Recall	ROC_AUC_Score	F1_score
LogisticRegression	0.000000	0.000000	49.893617	0.000000
KNeighborsClassifier	15.789474	14.285714	55.440729	15.000000
DecisionTreeClassifier	16.666667	19.047619	57.396150	17.777778
RandomForestClassifier	0.000000	0.000000	49.574468	0.000000
BernoulliNB	16.666667	9.523810	53.698075	12.121212
GradientBoostingClassifier	0.000000	0.000000	49.361702	0.000000
XGBClassifier	25.000000	4.761905	52.061803	8.000000
LightGBM	50.000000	9.523810	54.549139	16.000000

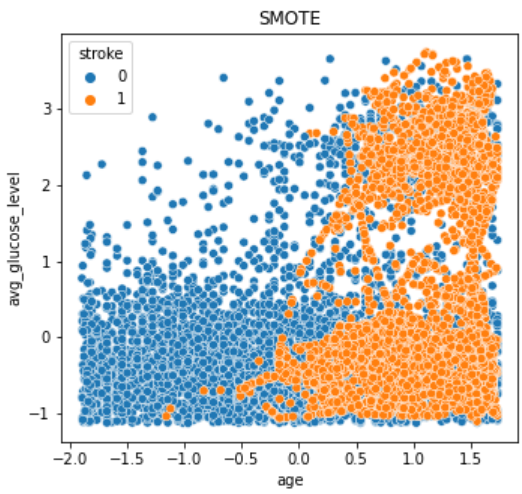
→ We can clearly see that how poor our models work as the dataset was highly imbalanced.

→ So, our first task is to make dataset balanced.

➢ Making Dataset balanced with the help of **Synthetic Minority Oversampling Technique (SMOTE)**

Smote Technique → Oversample the examples in the minority class. This can be achieved by simply duplicating examples from the minority class in the training dataset prior to fitting a model. This can balance the class distribution but does not provide any additional information to the model.



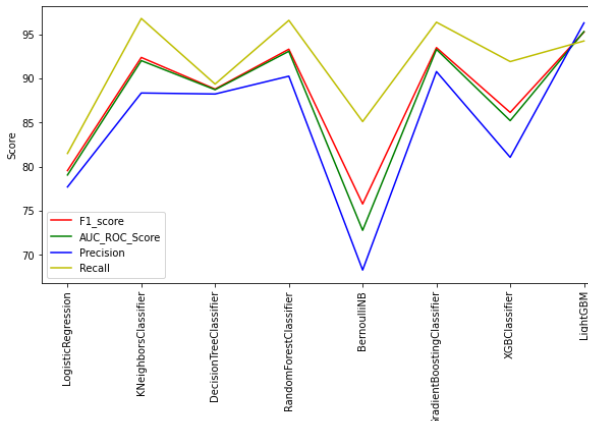


→ Dataset is oversampled using SMOTE Technique and the final dataset look like the right plot of the dataset.

→ Left image is the actual dataset.

➢ Modelling the balanced dataset using the same models used above

Model	Precision	Recall	ROC_AUC_Score	F1_Score
LogisticRegression	77.687627	81.489362	79.042553	79.543094
KNeighborsClassifier	88.349515	96.808511	92.021277	92.385787
DecisionTreeClassifier	88.235294	89.361702	88.723404	88.794926
RandomForestClassifier	90.258449	96.595745	93.085106	93.319630
BernoulliNB	68.259386	85.106383	72.765957	75.757576
GradientBoostingClassifier	90.781563	96.382979	93.297872	93.498452
XGBClassifier	81.050657	91.914894	85.212766	86.141575
LightGBM	96.304348	94.255319	95.319149	95.268817



→Selecting models having high Recall Score and having high ROC_AUC score. Hence, RFC, DTC, KNN, GBC, XGB, LGBM are selected. So, these models are now tuned by the hypermeters which is followed by 10-fold cross validation.

➔ 10 – Fold Cross Validation Result:
From the table on the left side that KNN and DTC do not work very well, Hence, we would take the rest 4 models for the final classification as the ROC_AUC score received is good to take.

Model	Cross Validation AUC ROC Score
RandomForestClassifier	0.978604
KNeighborsClassifier	0.906888
DecisionTreeClassifier	0.914198
GradientBoostingClassifier	0.978469
XGBClassifier	0.958511
LightGBM	0.935106

➢ Justification why these classifiers yield good results

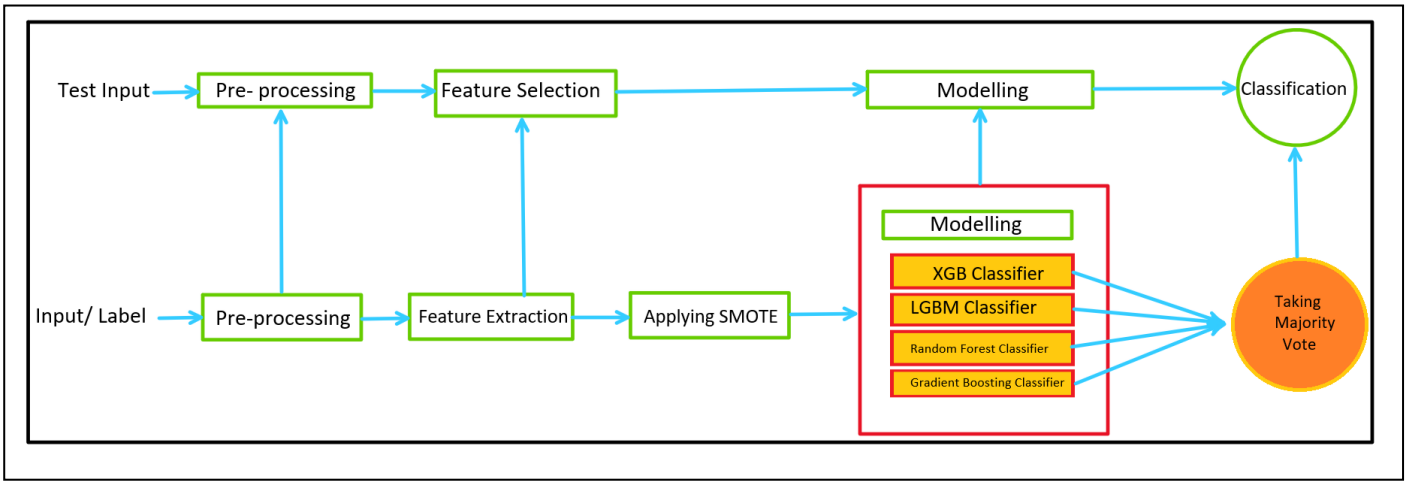
So, all the models selected uses ensemble learning technique where RFC uses bagging to give accuracy. In RFC, bagging is just like training a bunch of individual decision trees in a parallel way. Other is boosting technique which the other 3 classifiers uses and it's about training a bunch of individual models in sequential manner by learning from the mistakes done by the previous models.

➢ Training neural network and checking validation ROC_AUC_Score

We have made a neural network using Pytorch library having 7 hidden layers and used ReLU as the activation function between hidden layers and SoftMax at the last layer. The learning rate kept was 0.1 and number of epochs as 1000 but we have only received about 89 % of score on it, so we dropped to use neural network in the final classification.

➔ Final Prediction is received by taking the majority vote of the above chosen classifiers.

➢ END -TO – END PIPELINE FOR THE GIVEN DATASET



The final ROC_AUC score for the test data received is 95.4671%.

➢ Deployed the model and created the Web Application | Web-app

We have deployed the above model and created a web app of the same where user can input the parameters asked example gender, age etc. Based on the input parameters received, model predicts the probability of getting stroke and not getting stroke. If probability is greater than half then it predicts that person might suffer from stroke. User can even see the probability of getting and not getting stroke. We have used streamlit library to do for the same.

➢ Contribution of members

→ Shrutayu Aggarwal (B20CS096): Pre- processing, exploratory data analysis, Feature Engineering, modelling with SMOTE and metric evaluation, report, web app.

→ Priyanshu Jain (B20CS044): Neural Network, Hyper Tuned parameters, pipeline diagram, report, web app.

→ Pratul Singh (B20CS095): modelling without SMOTE, n-fold cross validation, pipeline, report, web app

GitHub repository link of the project